VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Jukebox

2011 **Bc. Michal Svatuška, DiS.**

Zadávací list

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 15.8.2011 .....

Rád bych na tomto místě poděkoval všem, kteří mě jak při tvorbě diplomové práce, tak i během samotného studia podporovali, zejména mým rodičům.

Abstrakt

Tato diplomová práce popisuje návrh, implementaci a spojení technologií jak HW tak i SW potřebných pro sestavení a provozování výsledného multiplatformního jukeboxu.
V práci je provedena analýza po HW i SW stránce, popis vybraných technologií a zdůvodnění tohoto výběru.
HW nároky jsou dány jednak zařízením samotným a také především tím, že je jukebox koncipován jako dotykově ovládané zařízení:

- Elektronický mincovník pro příjem mincí.
- Dotykové LCD.
- IrDA dálkové ovládání.

SW nároky jsou dány požadavkem na multiplatformnost řešení a preferencí open source technologií. Open source technologie jsou preferovány z mnoha hledisek, ať už jsou to hlediska kvalitativní, filozofické nebo finanční.
Preference open source technologií určuje také operační systém, na kterém bude výsledné zařízení prezentováno, a tím je GNU/Linux.
Podrobný popis technologií a konkrétních řešení je popsán v kapitolách o vybraném HW a SW a dále v kapitole o podrobném popisu implementace jednotlivých funkčních celků celého výsledného zařízení.

Klíčová slova

Jukebox, GNU/Linux, Linux, Java, MySQL, Mincovník, GStreamer, multimedia, framework, JNI (Java Native Interface), OSA (Ochranný svaz autorský), LibreOffice.

Abstract

This diploma work describes the design, implementation and combination of technologies both HW and SW needed to build and operate cross-platform jukebox.
In the diploma work is done analysis of the hardware and software sides and description and justification of selected technologies.
HW requirements are determined by device itself and especially because jukebox is designed as a touch-controlled device:

- Coin selector for acceptance coins.
- Touch screen LCD.
- IrDA remote control.

SW requirements are determined by the requirements for a cross-platform solutions and preferences of open source technologies. Open source technologies are preferred from many reasons, e.g. qualitative, philosophical or financial.
Preference for open source technology is also a determining factor for choice operating system on which the final product will be presented. And this operating system is GNU/Linux.
A detailed description of selected technologies and solutions are described in the chapters about selected hardware and software, and later in the chapter about the detailed description of the implementation of the functional parts of the final product.

Key words

Jukebox, GNU/Linux, Linux, Java, MySQL, Coin Selector, GStreamer, multimedia, framework, JNI (Java Native Interface), OSA (Copyright Protection Association), LibreOffice.

Seznam použitých symbolů a zkratek

- OSS – Open-source software (počítačový software s otevřeným zdrojovým kódem).
- GNU – Je rekurzivní zkratka GNU is Not Unix (GNU Není Unix) projektu na podporu vývoje svobodného software.
- GNU/Linux – GNU software + Linux kernel = varianty operačního systému GNU. GNU systémy jsou tzv. UNIX-like a neobsahují žádný originální kód Unixu.
- SSH – (Secure Shell) zabezpečený komunikační protokol využívající TCP/IP pro komunikaci v počítačových sítích.
- SW – software.
- HW – hardware.
- GUI – Graphical user interface.
- IrDA – Komunikační infračervený port vytvořený konsorciem IrDa (Infrared Data Association).
- UART – Universal Asynchronous Receiver/Transmitter.
- JNI – Java Native Interface.
- JNA – Java Native Access.
- LAF – Look and Feel (vzhled a chování).
- PC – Personal computer (osobní počítač).
- OSA – Ochranný svaz autorský.
- CZK – Koruna česká (měna).
- EUR – Euro (měna).
- LIRC – Linux Infrared Remote Control.
- HTML – HyperText Markup Language (hypertextový značkovací jazyk).
- FK – cizí klíč (foreign key).
- PK – primární klíč (primary key).
- aj. – a jiné.
- atd. – a tak dále.
- db – databáze.
- např. – například.
- tzv. – takzvaný/takzvaně.

Obsah

Obsah

Seznam tabulek

Seznam obrázků

Seznam příkladů kódů a konfiguračních souborů

1 Úvod

Mnohokrát jsem se setkal se situací kdy si majitel stěžoval na jukebox, který již zastaral, generoval nespočet chyb, popřípadě již dosluhoval i hardware. Výrobce stávající řešení již nepodporoval, popřípadě již vůbec neexistoval.

Na základě této zkušenosti se zrodila primární myšlenka a cíl této diplomové práce, vytvořit komplexní multiplatformní řešení jukeboxu s možností provozovat toto řešení s mírnými hardwarovými úpravami i na stávajících HW řešeních.

V prvních kapitolách práce je popsána provedená specifikace HW a SW potřeb na sestavení daného zařízení s následnou volbou konkrétních technologií a zdůvodněním této volby. Dále je provedena datová a funkční analýza, ze kterých následně vychází vlastní implementace softwarové části jukeboxu.

V následujících kapitolách práce je popisován implementace a spojení technologií jak HW tak i SW potřebných pro sestavení a provozování výsledného multiplatformního jukeboxu. Podrobný popis technologií a konkrétních řešení je popsán v kapitolách o vybraném HW a SW a dále v kapitole o podrobném popisu implementace jednotlivých funkčních celků celého výsledného zařízení.

Z hlediska softwarového pohledu je zde provedena analýza a implementace 3 hlavních částí:

- Databázová analýza a implementace konkrétního databázového řešení, zaměřujícího se na databázi jako na efektivní úložiště informací o fyzických datech přehrávaných a využívaných jukeboxem.
- Analýza a implementace softwarového řešení zaměřujícího se na komunikaci s hardwarovými periferiemi.
- Analýza a implementace softwarového řešení pro komunikaci s databází, multimediálním frameworkem a se samotným GUI, které zajišťuje intuitivní komunikaci a ovládaní pro uživatele.

Celé výsledné zařízení je koncipováno jako moderní zařízení s možností připojení na internet a využitím komplexní vzdálené správy, a to jak z hlediska funkčnosti a potencionálních servisních zásahů, tak i z hlediska správy datového obsahu přehrávaného a využívaného jukeboxem.

2 Specifikace požadavků

2.1 Cíl práce

Cílem diplomové práce je vytvoření komplexního řešení multiplatformního jukeboxu. K vytvoření jsou potřeba technologie jak HW tak i SW, přičemž spojení a implementace těchto technologií je hlavní náplní práce.

HW nároky jsou dány jednak zařízením samotným, ale také především tím, že je jukebox koncipován jako dotykově ovládané zařízení:

- Elektronický mincovník pro příjem mincí.
- Dotykové LCD.
- IrDA dálkové ovládání.

SW nároky jsou dány požadavkem na multiplatformnost řešení a preferencí open source technologií. Open source technologie jsou preferovány z mnoha hledisek, ať už jsou to hlediska kvalitativní, filozofické či finanční.

Preference open source technologií určuje také operační systém na kterém bude výsledné zařízení prezentováno a tím je GNU/Linux.

2.2 Seznam funkčních požadavků

1. Zpracování údajů o akceptované minci a navýšení dostupné finanční hotovosti využitelné pro následné přehrávání muziky.
2. Umožnit uživateli ovládat mincovník přes intuitivní GUI optimalizované pro dotykové ovládání.
3. Zobrazení a sortování alb dle různých parametrů a zobrazení podrobností aktuálně vybraného alba (songy daného alba, obal alba).
4. Zobrazení podrobností vybraného songu z aktuálně vybraného alba.
5. Přidání songu do fronty přehrávané muziky.
6. Přehrávání muziky z fronty přehrávané muziky.
7. Přidání songu do fronty přehrávané muziky pro přehrání jako karaoke, pokud tuto možnost daný song nabízí (dostupnost titulků).
8. Přehrávání muziky jako karaoke z fronty přehrávané muziky.
9. Přehrávaný song vložit do databáze statistik a umožnit uživateli prohlížet statistiky v přehledném GUI.
10. Zobrazit podrobnosti o vybraném songu, albu ze statistik.

2.3 Evidované údaje o albu a songu

Povinné a nepovinné údaje o albu a songu, které je nutné evidovat.

2.3.1 Povinné údaje o albu

- Název alba
- Rok vydání
- Songy
- Statistiky

2.3.2 Nepovinné údaje o albu

- Obal alba

2.3.3 Povinné údaje o songu

- Název songu
- Příslušnost k albu
- Pořadí na albu
- Karaoke (ano/ne)
- Žánr

2.3.4 Nepovinné údaje o songu

- Autor muziky
- Autor textu

2.4 Požadavky na HW

- Elektronický mincovník pro příjem mincí
- Dotykové LCD
- IrDA dálkové ovládání

2.5 Požadavky na SW

- Multiplatformnost řešení
- Preference open source technologií
- Optimalizace pro dotykové ovládání
- Statistiky přehrávaných songů

2.6 Vztahy mezi evidovanými údaji

SongHaveCopywriter – song může mít přiděleného autora textu.

SongHaveComposer – song může mít přiděleného autora muziky.

SongHaveInterpreter – song má přiděleného interpreta.

SongHaveGenre – song má přidělený žánr.

SongHaveAlbum – song je přidělen do alba.

SongPlayQueue – song je přidán do fronty pro přehrávání.

SongPlayStatistics – přehrávaný song z fronty pro přehrávání je přidán do statistiky přehrávané muziky.

3 Analýza

Samotná analýza je rozdělena na dvě hlavní části.

- Datová analýza – má za úkol vytvořit databázový systém, který bude co nejefektivnější z hlediska množství ukládaných dat a rychlosti práce s nimi.
- Funkční analýza – má za úkol vytvořit co nejefektivnější klientský systém z hlediska obsluhy požadavků, vznikajících užíváním systému uživateli.

Obě dvě části jsou stejně důležité, a proto se musí analýza provést velice pečlivě s důrazem na možnost vzniku chyb. Chyby vzniklé při špatné analýze se mohou velice těžce projevit při implementaci v podobě pozdějších velice náročných úprav.

3.1 Datová analýza

3.1.1 Lineární zápis

Lineární zápis reprezentuje jednotlivé tabulky a jejich atributy. Celkově je db tvořena 14ti tabulkami.

PK – (primary key) primární klíč je hlavní klíč a hlavní index tabulky. Každá tabulka musí mít primární klíč, který identifikuje jednotlivé záznamy tabulky. Primární klíč nesmí být nulový, obvykle je jím celočíselný datový typ, který má nastavenu vlastnost autoinkrement.

FK – (foreign key) cizí klíč je index tabulky a obvykle celočíselný datový typ. Cizí klíč slouží jako odkaz na primární klíč jiné tabulky a tím svazuje, přiřazuje daný záznam k jinému nadřazenému záznamu.

Značení v lineárním zápisu **tabulka** (primární klíč, ***cizí klíč***, atribut)

tab_songs (id_song, ***fid_album***, ***fid_genre***, ***fid_interpreter***, ***fid_composer***, ***fid_copywriter***, title_song, filepath_song, number_on_album_song, audio_video_song, karaoke_song)

tab_copywriter (id_copywriter, title_copywriter)

tab_composer (id_composer, title_composer)

tab_interpreter (id_interpreter, title_interpreter)

tab_genre (id_genre, title_genre)

tab_album (id_album, title_album, year_album, cover_album)

tab_play_queue (id_play_queue, ***fid_song***, timestamp_play_queue, play_as_karaoke_play_queue)

tab_play_statistics (id_play_statistics, ***fid_song***, timestamp_play_statistics, play_as_karaoke_play_statistics)

tab_money (variable_name_money, value_money)

3.1.2 Popis vztahů mezi tabulkami

SongHaveCopywriter (tab_songs, tab_copywriter)

SongHaveComposer (tab_songs, tab_composer)

SongHaveInterpreter (tab_songs, tab_interpreter)

SongHaveGenre (tab_songs, tab_genre)

SongHaveAlbum (tab_songs, tab_album)

SongPlayQueueHaveSong (tab_songs, tab_play_queue)

SongPlayStatisticsHaveSong (tab_songs, tab_play_statistics)

3.1.3 Datový slovník

PK – primární klíč (primary key)

FK – cizí klíč (foreign key)

Index – I (I – je index)

Null – N (N – nulový)

Vlastnosti –autoincrement, unsignet (automaticky se zvětšuje velikost primárního klíče s přidávaným záznamem, nezáporná hodnota primárního klíče)

Charset celé db je nastaven na utf8_czech_ci

Všechny pole typu varchar a text mají nastaveno porovnávání (utf8_czech_ci)

tab_songs						
Atribut	**Typ(vel.)**	**Klíč**	**Index**	**Null**	**Vlastnosti**	**Popis**
id_song	int(10)	PK	I		unsignet autoincrement unique	Primární klíč tabulky tab_songs
fid_album	int(10)	FK	I		unsignet	Cizí klíč na tabulku tab_album
fid_genre	int(10)	FK	I		unsignet	Cizí klíč na tabulku tab_genre
fid_interpreter	int(10)	FK	I		unsignet	Cizí klíč na tabulku tab_interpreter
fid_composer	int(10)	FK	I	N	unsignet	Cizí klíč na tabulku tab_composer
fid_copywriter	int(10)	FK	I	N	unsignet	Cizí klíč na tabulku tab_copywriter
title_song	varchar(100)		I			Titulek songu
filepath_song	text					Cesta k souboru songu na disku
number_on_album_song	int(2)				unsignet	Pořadové číslo songu na albu
audio_video_song	tinyint(1)					Parametr zda se jedná o audio nebo video klip
karaoke_song	tinyint(1)					Parametr zda je pro daný song k dispozici karaoke

Tabulka 1: Datový slovník – tab_songs

tab_copywriter						
Atribut	**Typ(vel.)**	**Klíč**	**Index**	**Null**	**Vlastnosti**	**Popis**
id_copywriter	int(10)	PK	I		unsignet autoincrement unique	Primární klíč tabulky tab_copywriter
title_copywriter	varchar(200)					Název autora textu

Tabulka 2: Datový slovník – tab_copywriter

tab_composer						
Atribut	**Typ(vel.)**	**Klíč**	**Index**	**Null**	**Vlastnosti**	**Popis**
id_composer	int(10)	PK	I		unsignet autoincrement unique	Primární klíč tabulky tab_composer
title_composer	varchar(200)					Název skladatele

Tabulka 3: Datový slovník – tab_composer

tab_interpreter						
Atribut	**Typ(vel.)**	**Klíč**	**Index**	**Null**	**Vlastnosti**	**Popis**
id_interpreter	int(10)	PK	I		unsignet autoincrement unique	Primární klíč tabulky tab_interpreter
title_interpreter	varchar(200)					Název skladatele

Tabulka 4: Datový slovník – tab_interpreter

tab_genre						
Atribut	**Typ(vel.)**	**Klíč**	**Index**	**Null**	**Vlastnosti**	**Popis**
id_genre	int(10)	PK	I		unsignet autoincrement unique	Primární klíč tabulky tab_genre
title_genre	varchar(100)					Název žánru

Tabulka 5: Datový slovník – tab_genre

tab_album						
Atribut	**Typ(vel.)**	**Klíč**	**Index**	**Null**	**Vlastnosti**	**Popis**
id_album	int(10)	PK	I		unsignet autoincrement unique	Primární klíč tabulky tab_album
title_album	varchar(200)					Název alba
year_album	int(4)					Rok vzniku alba
cover_album	varchar(300)			N		Cesta k obalu alba

Tabulka 6: Datový slovník – tab_album

tab_play_queue						
Atribut	**Typ(vel.)**	**Klíč**	**Index**	**Null**	**Vlastnosti**	**Popis**
id_play_queue	int(10)	PK	I		unsignet autoincrement unique	Primární klíč tabulky tab_play_queue
fid_song	int(10)		I			Cizí klíč na tabulku tab_song
timestamp_play_queue	timestamp		I			Časové razítko kdy je vložen song do fronty
play_as_karaoke_play_queue	tinyint(1)					Přehrávat jako karaoke – ano/ne

Tabulka 7: Datový slovník – tab_play_queue

tab_play_statistics						
Atribut	**Typ(vel.)**	**Klíč**	**Index**	**Null**	**Vlastnosti**	**Popis**
id_play_statistics	int(10)	PK	I		unsignet autoincrement unique	Primární klíč tabulky tab_play_statistics
fid_song	int(10)		I			Cizí klíč na tabulku tab_song
timestamp_play_statistics	timestamp		I			Časové razítko kdy je vložen song do statistik
play_as_karaoke_play_statistics	tinyint(1)					Song byl přehrán jako karaoke – ano/ne

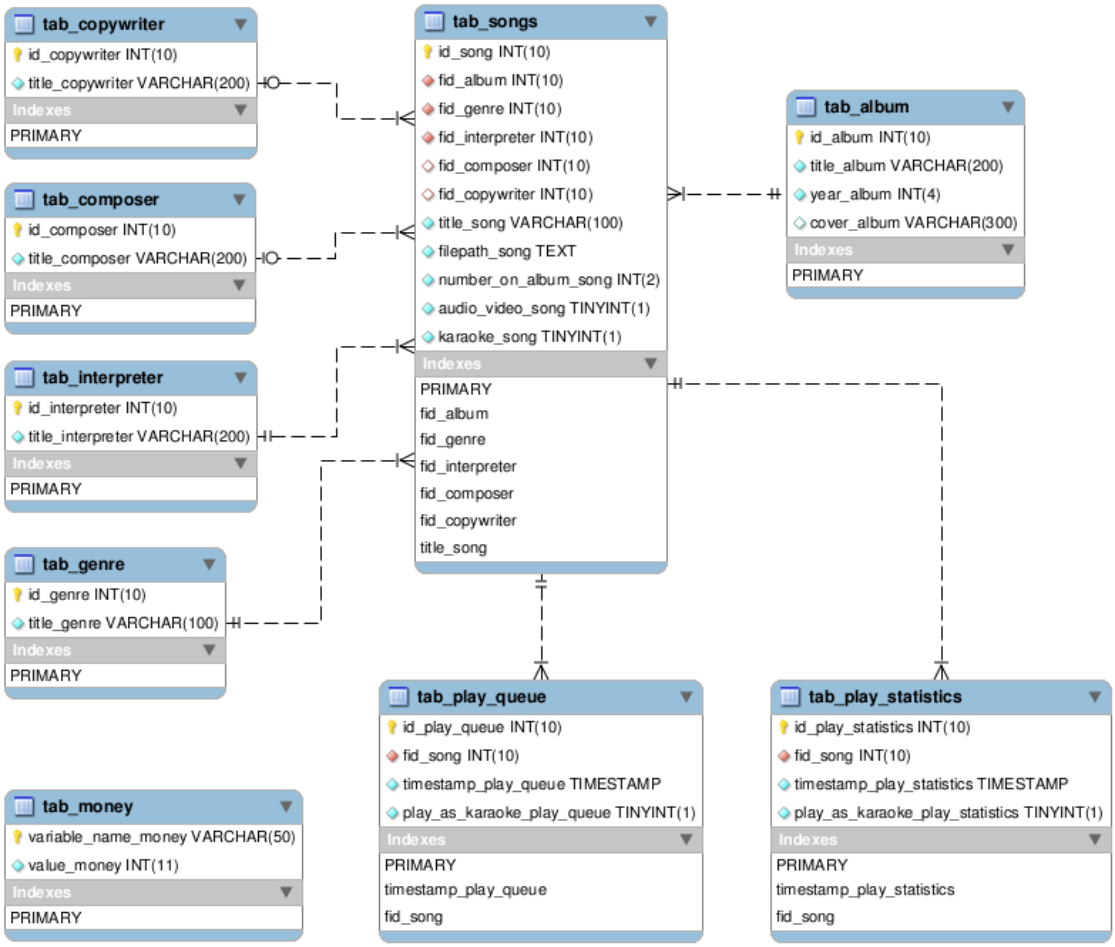
Tabulka 8: Datový slovník – tab_play_statistics

tab_money						
Atribut	**Typ(vel.)**	**Klíč**	**Index**	**Null**	**Vlastnosti**	**Popis**
variable_name_money	varchar(50)	PK	I		unsignet autoincrement unique	Primární klíč tabulky tab_money
value_money	int(11)					Hodnota konfigurační proměnné

Tabulka 9: Datový slovník – tab_money

3.1.4 ER diagram

ER diagram zobrazuje celou strukturu databáze se všemi relacemi, kterými jsou jednotlivé tabulky spojeny.



Obrázek 1: ER diagram

4 Funkční analýza

Ve funkční analýze bude popsána funkční část klientského programu jukeboxu v podobě Use Case (případ užití) diagramů a jejich podrobného textového popisu.

Jako první Use Case bude zobrazen náhled na celý systém a poté budou podrobně popsány jednotlivé subsystémy jukeboxu.

Aktor – vystihuje roli uživatele nebo jiného systému při interakci s naším systémem.

UCXXX Název – jednotlivé Use Case budou značeny v tomto tvaru (XXX=číselné označení např. UC001 Název Use Case).

Toky událostí v užitích:

- *Základní tok událostí* – v základním toku událostí je umístěn maximálně stručný a přitom kompletní postup při realizaci případu užití. V základním toku se nesmí objevit žádné větvení toku ani žádné podrobné informace. Tok případů užití, s nimiž je spojen popisovaný případ užití relací <<Include>> nebo <<Extend>>, může být do hlavního toku vložen zapsáním slova <<Include>> nebo <<Extend>> následovaném plným názvem inkriminovaného případu užití.
- *Alternativní tok událostí* – v alternativním toku jsou zapsány všechny odchylky od lineárního toku událostí (podmíněné kroky postupu, opakování kroků – cykly).
- *Rozšiřující tok událostí* – rozšiřující tok obsahuje všechny relevantní podrobnosti k bodům postupu v hlavním toku.

Více v [1], [2].

4.1 Systém a subsystémy

Jukebox se skládá z několika subsystémů. Jednotlivé subsystémy budou rozebrány ve funkční analýze pomocí výše zmiňovaných Use Case diagramů a jejich podrobného textového popisu.

Systém:

- GSJukeBox

Subsystémy:

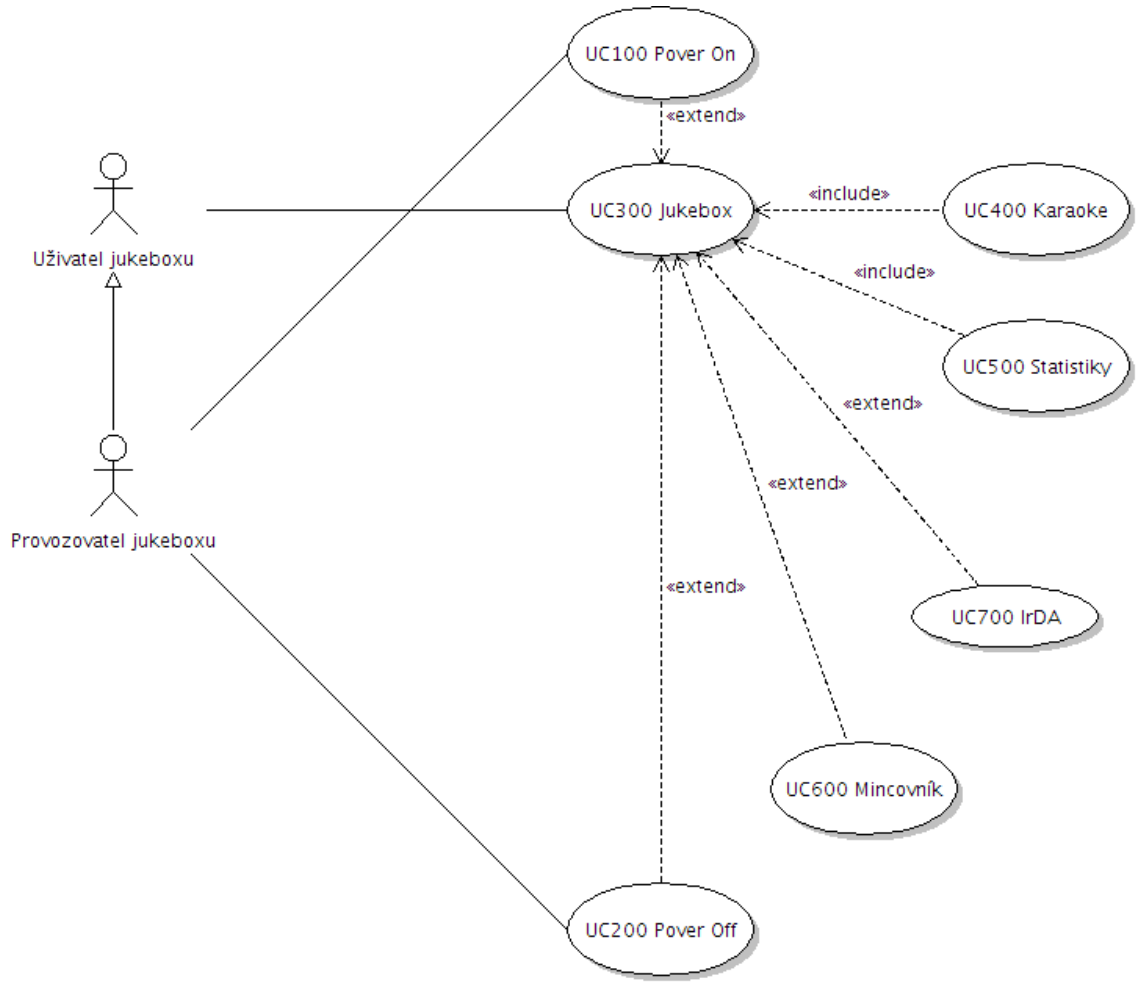
- Jukebox
- Karaoke
- Statistiky
- Mincovník
- IrDA

4.2 Systém – GSJukeBox

Celý systém je postavený na multimediálním frameworku GStreamer, a proto získal název GSJukeBox.

Základní pohled na celý systém obsahuje 7 případů užití:

- UC100 Pover On
- UC200 Pover Off
- UC300 Jukebox
- UC400 Karaoke
- UC500 Statistiky
- UC600 Mincovník
- UC700 IrDA



Obrázek 2: Use Case – Systém GSJukeBox

4.3 UC300 Jukebox

Základní tok událostí

1. Uživatel vybere parametry pro selektování alb.
2. Uživatel vybere konkrétní album.
3. Uživatel vybere konkrétní song vybraného alba.
4. Uživatel vybere volbu „Hrej“a tím zobrazí dialog pro potvrzení přehrání.
5. Uživatel vybere volbu „Přehrát“.
6. Je odečten příslušný objem peněz za přehrávaný song z peněz aktuálně dostupných pro přehrávání.
7. Song je přidán do fronty přehrávaní.
8. Song se začne přehrávat.
9. Přehrávaný song je vložen do statistik přehrávaných songů.
10. Přehrávaní je ukončeno.
11. Kontrola zda je ve frontě další song pro přehrávání a pokud ano, následuje bod 8.

Alternativní tok událostí

Body 1-10 – Uživatel může volbou „Statistiky“ zobrazit dialog statistiky přehrávaných songů << include >> UC500 Statistiky.

Bod 3 – po bodu 3 lze využít << include >> UC301 Zobrazení podrobností o songu.

Bod 4 – V případě nedostatku peněz aktuálně dostupných pro přehrávání, je o tomto uživatel informován prostřednictvím informačního dialogu.

Bod 5 – dialog lze zavřít pomocí volby „Zrušit“, nebo lze song přehrát jako karaoke pokud tuto volbu daný song nabízí << include >> UC302 Přehrát jako karaoke.

Bod 7 – v případě, že je ve frontě alespoň jeden song, může uživatel volbou „Podrobnosti“ zobrazit o tomto songu podrobnosti << include >> UC301 Zobrazení podrobností o songu.

Bod 8 – v případě, že je aktuálně přehrávaný jiný song, tak právě přidaný song čeká ve frontě až na něho přijde řada s přehráváním.

Bod 8 – pokud je na vrcholu fronty song, který má být přehrán jako karaoke << include >> UC400 Karaoke.

Rozšiřující tok událostí

Bod 1 – selektovat lze dle názvu alba, názvu interpreta, názvu songu, názvu žánru. Každý z těchto parametrů lze doselektovat o výběr pouze položek začínajících na konkrétní písmeno.

Uživatel může vhazovat mince << extend >> UC600 Mincovník a používat dálkové ovládání k ovládání hlasitosti jukeboxu << extend >> UC700 IrDA.

4.3.1 UC301 Zobrazení podrobností o songu

Základní tok událostí

1. Uživatel volbou „Podrobnosti“ otevře dialog podrobnosti o songu.
2. Uživatel volbou „Zavřít“ zavře dialog podrobnosti o songu.

Alternativní tok událostí

Rozšiřující tok událostí

Uživatel může vkládat mince << extend >> UC600 Mincovník a používat dálkové ovládání k ovládání hlasitosti jukeboxu << extend >> UC700 IrDA.

4.3.2 UC302 Přehrát jako karaoke

Základní tok událostí

1. Je odečten příslušný objem peněz za přehrávaný song jako karaoke z peněz aktuálně dostupných pro přehrávání.
2. Song je přidán do fronty přehrávaní s označením, že bude song přehrán jako karaoke.
3. << include >> UC400 Karaoke.

Alternativní tok událostí

Bod 3 – v případě, že je přehrávaný jiný song, tak aktuálně přidaný song čeká ve frontě až na něho přijde řada z přehráváním.

Rozšiřující tok událostí

Uživatel může vkládat mince << extend >> UC600 Mincovník a používat dálkové ovládání k ovládání hlasitosti jukeboxu << extend >> UC700 IrDA.

4.4 UC400 Karaoke

Základní tok událostí

1. Přehrávaný song jako karaoke je načten, jukebox je přepnut do módu karaoke a vyčkává na start karaoke uživatelem.
2. Uživatel volbou „Spust“ spustí přehrávání songu jako karaoke.
3. Přehrávaný song jako karaoke je vložen do statistik přehrávaných songů.
4. Song přehráván jako karaoke je přehráván.
5. Přehrávaní je ukončeno.
6. Pokračování UC300 Jukebox – bod 11.

Alternativní tok událostí

Bod 1 – Uživatel může volbou „Statistiky“ zobrazit dialog statistiky přehrávaných songů << include >> UC500 Statistiky.

Rozšiřující tok událostí

Uživatel může vkládat mince << extend >> UC600 Mincovník a používat dálkové ovládání k ovládání hlasitosti jukeboxu << extend >> UC700 IrDA.

4.5 UC500 Statistiky

Základní tok událostí

1. Uživatel vybere volbu „Statistiky“.
2. Uživatel vybere parametry, dle kterých lze selektovat statistiky.

Alternativní tok událostí

Rozšiřující tok událostí

Bod 2 – uživatel může zobrazit podrobnosti o vybraném songu statistik volbou „Podrobnosti“ << include >> UC301 Zobrazení podrobností o songu.

Uživatel může vkládat mince << extend >> UC600 Mincovník a používat dálkové ovládání k ovládání hlasitosti jukeboxu << extend >> UC700 IrDA.

4.6 UC600 Mincovník

Základní tok událostí

1. Uživatel vhodí minci do mincovníku.
2. Mince je akceptována.
3. Je navýšena finanční hotovost pro přehrávání o hodnotu akceptované mince.

Alternativní tok událostí

Bod 2 – Mince není akceptována a je vrácena mincovníkem zpět.

Rozšiřující tok událostí

4.7 UC700 IrDA

Základní tok událostí

1. Uživatel stiskne na dálkovém ovladači tlačítko pro zesílení zvuku.
2. Je zesílen zvukový systém operačního sytému.

Alternativní tok událostí

Bod 1 – Uživatel stiskne tlačítko pro zeslabení zvuku.

Bod 2 – Je zeslaben zvukový systém operačního systému.

Rozšiřující tok událostí

5 Zvolené HW řešení a technologie

HW nároky jsou dány především zařízením samotným (mincovník, IrDA) a tím, že je jukebox koncipován jako dotykově ovládané zařízení (dotykové LCD).

5.1 Mincovník

Zvolený mincovník je výrobek italské firmy „Alberici S.p.A.“ zabývající se výrobou zařízení pro bankovní sektor, směnu peněz, komponenty pro hrací automaty atd.

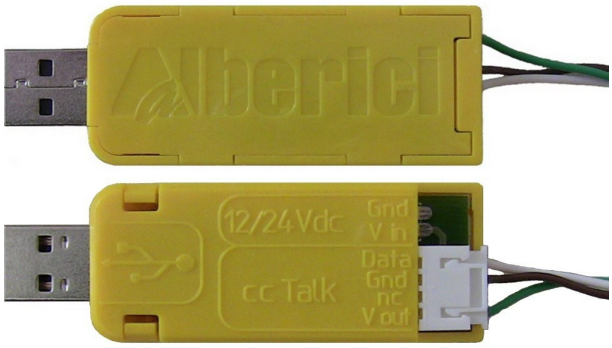
Konkrétně zvolený model mincovníku – **AL55 ccTalk model S**.

Jednotlivé modely rozlišené písmeny se odlišují dle různých parametrů:

- Velikost předního panelu.
- Umístění pozice pro příjem mincí.
- Umístění pozice pro vracené neakceptované mince.
- Umístění pozice pro vyústění akceptovaných mincí.
- atd.

Popis a zdůvodnění volby:

- Mincovník komunikuje pomocí moderního protokolu ccTalk.
- Možnost využít USB interface pro přímé připojení k PC (programování, komunikace, napájení).
- Plně programovatelný mincovník (CZK, EUR atd.).
- Výborná komunikace se zástupcem firmy.



Obrázek 3: USB interface

Obrázek 4: Mincovník AL55 ccTalk model S

5.2 IrDA

Řešení IrDA ovladače je založeno na systému LIRC. Jako HW lze využít jakýkoliv ze standardně prodávaných infrared receiverů, které mají drivery pro potřebné platformy operačního systému.

5.3 Dotykové LCD

Na dotykové LCD nejsou žádné zvláštní požadavky. Může být zvoleno jakékoliv ze standardně prodávaných LCD, které disponuje drivery pro potřebné platformy operačního systému.

5.4 Počítač

Parametry počítače, na kterém bude daný systém provozován jsou dány především minimálními požadavky operačního systému, který bude zvolen pro provozování jukeboxu.

Konkrétně zvolený model počítače – **Nettop ALFA Starter Micro** zakoupený od firmy ALFA COMPUTER a.s.

Popis a zdůvodnění volby:

- Malá spotřeba, s tím spojená malá produkce tepelné energie a malé nároky na chlazení.
- Malé kompaktní řešení.
- Velký počet USB.
- Standardní vývod zvukové karty.

Základní technické parametry:

- Bez OS, zdroj – napájecí adaptér.
- Chipset Intel NM10.
- Intel Atom D510, 1.66 GHz (dual-core).
- 2 GB DDR2 800 MHz Kingston SO-DIMM.
- Pevný disk 320 GB, 2.5", 5400 ot./min., SATA 3 Gb/s, 8 MB cache.
- Grafická karta integrovaná Intel GMA 3150.
- Zvuková karta HD Audio Realtek ALC662.
- Síťová karta 10/100/1000 Mbps.
- USB 2.0 4x vzadu, 2x vpředu.
- Spotřeba v klidu 22 W.
- Spotřeba při zátěži 27 W.

Obrázek 5: Nettop ALFA Starter Micro

6 Zvolené SW řešení a technologie

SW nároky jsou dány požadavkem na multiplatformnost řešení a preferencí open source technologií. Open source technologie jsou preferovány z mnoha hledisek, ať už jsou to hlediska kvalitativní, filozofické nebo finanční.

Preference open source technologií určuje také operační systém, na kterém bude výsledné zařízení prezentováno, a tím je GNU/Linux.

6.1 Java

Jako programovací jazyk byla zvolena Java v aktuální verzi 1.6.0_21-b06. Java je moderní vysokoúrovňový programovací jazyk maximálně splňující požadavky na multiplatformnost výsledného systému.

Popis a zdůvodnění volby:

- Moderní vysokoúrovňový programovací jazyk.
- Maximálně splňující požadavky na multiplatformnost výsledného systému.
- Velmi dobrá integrace s ostatními technologiemi (mysql, gstreamer atd.).
- Výborné GUI toolkity SWT a Swing.
- Dostupnost výborných IDE (Eclipse, Netbeans atd.).
- Dostupnost výborného GUI návrháře WindowBuilder Pro.
- Technologie properties, log4j.
- Obrovský „Java User Land“.
- Neskutečné množství technologií (např. od Apache Software Foundation).
- atd.
- Více v [3], [4].

6.2 GUI – SWT vs Swing

Dva nejpoužívanější GUI toolkity používané v javě jsou SWT a Swing. Každý z těchto toolkitů má své výhody i nevýhody, ale oba jsou výbornou volbou pro tvorbu GUI.

6.2.1 SWT

Standard Widget Tookit (SWT) je knihovna grafických uživatelských prvků pro platformu Java. Původně byla tato knihovna vyvinuta firmou IBM a nyní je ve správě nadace Eclipse Foundation jako součást projektu Eclipse IDE. Knihovna SWT je napsaná v Javě. K vykreslování prvků GUI používá nativní knihovny operačních systémů prostřednictvím JNI (Java Native Interface), a to obdobným způsobem jako programy psané s použitím nativního API operačního systému.

Výhoda:

- SWT má jednoznačnou výhodu v nativním chování a nativním vzhledu na každé platformně operačního systému.
- Změna v grafické podobě API (Windows XP, Windows 7), neznamená úpravu projektu. Změny v podobě aplikace jsou zajištěny automaticky (jsou volány aktuální API daného operačního systému).

Nevýhoda:

- Nutnost přiložit pro každou platformu specifický **swt.jar** soubor.

Více v [5].

6.2.2 Swing

Swing je knihovna uživatelských prvků na platformě Java. V roce 1997 zavrhl Sun vývoj GUI toolkitu AWT a započal vyvíjet GUI toolkit Swing. Jako základ sloužily Internet Foundation Classes (IFC), vyvinuté Netscape Communications Corporation na konci roku 1996, jako nástroj pro tvorbu grafického uživatelského rozhraní pro Javu. V roce 1997 došlo ke spojení IFC a dalších technologií od Netscape a Sun Microsystems do Java Foundation Classes (JFC), jehož je Swing součástí. Swing je vyvíjen jako kombinace AWT, IFC a dalších technologií a je nedílnou součástí Java SE od verze 1.2.

Výhoda:

- „Pure Java“ bez nutnosti tvorby balíků pro jednotlivé platformy operačních systémů.
- Pomocí Swing LAF (look and feel), lze dobře skinovat výsledný vzhled celé aplikace.

Nevýhoda:

- V případě nativního vzhledu ne vždy dokonalý výsledek (nedostatky nativního GTK+ vzhledu atd.).

Více v [6].

6.2.3 Zvolený GUI toolkit

Pro klasickou desktop aplikaci s potřebou nativního vzhledu by bylo zcela určitě vhodnější zvolit GUI toolkit SWT. Potřeby jukeboxu jsou však poněkud jiné. Více než plně nativní vzhled, zde vyvstává potřeba jednoduše graficky celou aplikaci skinovat. Skinovat celou aplikaci lze nejlépe za pomoci vlastního Swing LAF, a proto byl jako GUI toolkit v tomto případě zvolen Swing.

Více v [6].

6.3 Properties

Technologie java properties je standardní a velice pohodlná technologie jak ukládat aplikační konfigurační parametry java aplikací. Jednotlivé záznamy se skládají z proměnné (key) a její hodnoty (value).

Popis a zdůvodnění volby:

- Přímo integrovaná technologie v Jave.
- Velice pohodlná práce.

Možné formáty key a value:

- key=value
- key = value
- key: value
- key value

Java obsahuje techniky pro pohodlné načtení souboru, jeho automatické rozparsování a následné pohodlné získávání jednotlivých hodnot (value) pomocí proměnné (key).

Více v [7].

```
Properties properties = new Properties();
FileInputStream fileInputStream = new FileInputStream("file.properties");
InputStreamReader inputStreamReader = new InputStreamReader(fileInputStream,
"UTF8");
properties.load(inputStreamReader);
String str = properties.getProperty("someKey");
```

Kód 1: Práce s properties

6.4 Log4J

Log4j je v současnosti standardní technologie využívaná pro logování java aplikací.

Log4J se skládá ze tří částí:

1. Logger
2. Appender
3. Layout

6.4.1 Logger

Logger je základní částí log4j, je zodpovědný za zachycení logovacích informací. K dispozici je 5 + 2 speciálních úrovní logování:

- DEBUG – využívané pro logování debug informací.
- INFO – využívané pro logování informativních zpráv.
- WARN – využívané pro logování potencionálně škodlivých či nebezpečných informací.
- ERROR – využívané pro logování chybových zpráv.
- FATAL – využívané k logování zpráv, které mohou mít fatální následky na aplikaci.

- ALL – reprezentuje logování všech 5ti předchozích úrovní.
- OFF – žádná z předchozích 5ti úrovní se neloguje.

6.4.2 Appender

Appender je zodpovědný za publikování log informací do cíle. Existuje mnoho různých appenderů lišících se dle cíle publikování informací (např. ConsoleAppender, FileAppender, TelnetAppender, SMTPAppender atd.).

6.4.3 Layout

Každý appender musí mít přidělený layout, který mu určí jak bude daný výstup formátovaný. Layout se tedy stará o formátování výstupu appenderu.

Existují tři základní typy layoutů:

1. HTMLLayout – formátuje výstup jako HTML tabulku.
2. PatternLayout – formátuje výstup prostřednictvím conversion pattern, značka v podobě %písmeno je nahrazena konkrétní hodnotou (např. %p - priorita zprávy).
3. SimpleLayout – základní jednoduchý způsob (úroveň logování, pomlčka, log zpráva).

6.4.4 Použití

Pro použití Log4J je nutné k aplikaci připojit knihovnu **log4j.jar**. Vše lze definovat jednak ve zdrojovém kódu a jednak v konfiguračním properties souboru. Výhodou properties souboru je, že není nutné rekompilovat aplikaci po každé provedené změně v konfiguraci log4j (logger, appender, layout).

Více v [8].

6.5 GStreamer

GStreamer je multimediální framework snažící se co nejvíce ulehčit práci programátorům tvořícím multimediální aplikace. Formou pluginů podporuje velké množství formátů (MP3, OGG Vorbis, FLAC, AAC, XviD, Theora, DV, H264 atd.) a výstupních rozhraní (PulseAudio, ALSA, OSS, ESD atd.).

GStreamer je hlavní multimediální framework v desktopovém prostředí GNOME, které jej obsahuje od verze GNOME 2.2 a podporuje využívání GStreamer ve všech GNOME/GTK+ programech. Jako backend ho používá velké množství programu jak ze samotného GNOME tak i konkurenčního KDE a dalších. (Totem, Exaile, Amarok, Kaffeine, Rhytmbox atd.).

GStreamer samotný je napsaný v jazyce C, ale je k dispozici obrovské množství napojení (binding) do jiných programovacích jazyků jako Python, Java atd. GStreamer je vydávaný pod licencí LGPL a jeho portace jsou k dispozici pro různé operační systémy.

6.6 Gstreamer-java

Gstreamer-java zajišťuje pomocí napojení (binding) připojení multimediálního frameworku GStreamer do jazyka Java. V současné době je k dispozici pro hlavní platformy operačních systémů (Linux, MacOSX a Windows). Samotné napojení je poskytováno přes technologii JNA (Java Native Access), která je spolu s technologii JNI (Java Native Interface) způsobem, jak z jazyka Java volat nativní programy napsané např. v jazyce C (běžící mimo JVM). Gstreamer-java je také jako GStreamer samotný vydávaný pod licencí LGPL.

Části gstreamer-java:

- **gstreamer-java.jar** – obsahující objekty a jejich metody, které jsou posléze napojeny prostřednictvím jna.jar na samotný GStreamer.
- **gstreamer-java-swt.jar** – implementace v případě využívání SWT.
- **jna.jar** – vlastní knihovna zajišťující napojení (binding) java metod na nativní GStreamer napsaný v jazyce C.

Více v [9], [10].

6.7 ccTalk

Sériový protokol ccTalk je využívaný v široké míře pro zprostředkování komunikace mezi stroji poskytujícími peněžní transakce jako jsou platební automaty, validátory a zásobníky peněz, hrací automaty, telefonní automaty, automaty pro prodej vstupenek a lístků atd.

Protokol byl vyvinut ve společnosti Coin Controls, nyní existující pod názvem Money Controls v severozápadní Anglii, inženýrem Andy Barsonem. První verze protokolu vznikla v roce 1996.

Protokol používá asynchronní přenos znaku snímků podobně jako protokol RS232. Hlavní rozdíl je v tom, že používá jeden datový vodič pro obousměrnou half-duplex (buď vysílá, nebo přijímá) komunikaci. Funguje na napětí TTL a "multi-drop", tj. periferní zařízení mohou být připojeny na společnou sběrnici a jsou logicky odděleny adresou zařízení. Každé zařízení na sběrnici musí mít jedinečnou adresu. Původní protokol využíval frekvenci 4800 baud, následně byla vydána standardizace na 9600 baud.

Nízká cena čipů a dostupnost čipů od mnoha výrobců spolu s existencí zařízení s možností běhu přes USB a neexistencí licenčních poplatků dělají z tohoto protokolu v dnešní době standard v této oblasti.

Popis a zdůvodnění volby:

- Nízká cena UART zařízení.
- Snadno srozumitelné struktury paketů (zpráv).
- Žádné licenční poplatky.

```
TX data = 002 000 001 245 008
002 = destination address
000 = zero data bytes
001 = source address
245 = command header
008 = checksum ( 002 + 000 + 001 + 245 + 008 = 256 = 0 mod 256 )
```

Kód 2: Příklad TX paketu

```
RX data = 001 013 002 000 067 111 105 110 032 065 099 099 101 112 116 111
114 022
001 = destination address
013 = 13 data bytes
002 = source address
000 = reply header
067…114 = ASCII for ‘Coin Acceptor’
022 = checksum ( sum of all packet bytes is zero )
```

Kód 3: Příklad RX paketu

Více v [11], [12].

6.8 LIRC

LIRC (Linux Infrared Remote Control) je balík, který umožňuje dekódovat a posílat infračervené signály z mnoha běžně používaných dálkových ovládání.

Jednotlivé operační systémy:

- Linux – lirc, jak již jeho název napovídá, pochází z linuxového světa a má plnou podporu na linuxových operačních systémech.
- MacOSX – současná verze funguje na systémech Darwin.
- Windows – WinLirc je port Lirc pro operační systém Windows.

Daemony lirc systému:

- **lircd** – hlavní daemon lirc systému, který dekóduje IR signály a poskytuje jednotné rozhraní pro klientské aplikace.
- **lircmd** – daemon překládající IR signály na události myši.
- **irexec** – daemon sloužící ke spouštění jiných aplikací.
- **irxevent** – daemon zasílající parametry jako eventy jednotlivým aplikacím jako klávesové zkratky atd.

Konfigurační soubory lirc systému:

- **lircd.conf** – konfigurační soubor daemona lircd, obsahuje nadefinované jednotlivé názvy tlačítek ovladače a jim přiřazené kódy.
- **.lircrc (lircrc)** – konfigurační soubor sloužící pro mapování akcí na jednotlivé názvy tlačítek ze souboru lircd.conf.
- **hardware.conf** – konfigurační soubor pro konfigurování argumentů lirc systému jako jsou input device, driver, moduly, cesta ke konfiguračnímu souboru lircd.conf atd.

Další pomocné programy:

- **irw** – program služící k ověření zda jsou tlačítka a jejich kódy spojeny s návěštími v podobně názvů buttonů v lircd.conf.
- **irrecord** – program, který je možný využít na získání všech validních názvů tlačítek a jejich kódů.

Více v [13].

Dále je uveden podrobnější popis některých částí, které budou již popisovány na konkrétním operačním systému, a tím je Linux.

6.8.1 lircd a lircd.conf

Nejprve je nutné zjistit pod jakým **eventX** pracuje náš IR receiver. Na linuxu lze jednoduše vypsat input devices, popřípadě využít grep s patřičným klíčovým slovem.

```
cat /proc/bus/input/devices
cat /proc/bus/input/devices | grep [klíčové slovo]
```

Kód 4: Zjištění eventX device

Dále je nutné vygenerovat konfigurační soubor lircd.conf, ve kterém jsou názvy tlačítek a jejich číselné kódy. Toto je možné udělat dvěma způsoby.

1. Je možné využít výborný nástroj **gnome-lirc-properties**, který vše vygeneruje sám.
2. Je možné využít nástroj **irrecord**.
   - irrecord --list-namespace – vypíše všechny validní názvy tlačítek.
   - irrecord – program postupně vypisuje jednotlivé názvy tlačítek a uživatel si stiskem tlačítka na ovladači k tomuto názvu tlačítka přiřadí číselný kód tlačítka na ovladači (např. irrecord -H dev/input -d /dev/input/eventX lircd.conf).

Správné přiřazení názvu butonu k fyzickému tlačítku přes číselný kód lze otestovat pomocí programu **irw**, jež vypisuje při stisknutí tlačítka na ovladači konkrétní přiřazený název tlačítka.

```
begin remote
      name  linux-input-layer
      bits  32
      begin codes
       BTN_BACK    0x10116
       KEY_MUTE    0x10071
       KEY_PLAY    0x100cf
       KEY_STOP    0x10080
      end codes
end remote
```

Kód 5: Příklad souboru lircd.conf

Daemona **lircd** je nutné spouštět pod uživatelem root a to např. při startu v /etc/rc.d/rc.local (lircd -H dev/input -d /dev/input/eventX)

6.8.2 lircrc

Soubor lircrc obsahuje mapování akcí na jednotlivé názvy tlačítek ze souboru lircd.conf.

Možnosti umístění:

- lokální umístění – ~/.lircrc v home adresáři uživatele
- globální umístění – /etc/lirc/lircrc (toto umístění má vyšší prioritu než jednotlivá umístění v home adresářích uživatelů)

Programy s integrovaným ovládáním přes lirc systém:

Mnoho aplikací má již podporu lirc integrovanou, nebo stačí načíst modul který slouží pro komunikaci s lirc systémem. V tomto případě postačuje pouze soubor lircrc, ve kterém se volá přímo daný program a jeho předdefinované parametry pro ovládání přes lirc systém.

```
begin
   prog = pulseaudio
   remote = *
   button = KEY_MUTE
   config = mute-toggle
end
```

Kód 6: Příklad souboru lircrc

Programy bez integrovaného ovládání přes lirc systém:

Pokud program, který chceme ovládat, nemá přímou podporu lirc systému, lze využít programy **irexec** a **irxevent.**

- **irexec** – daemon sloužící ke spouštění jiných aplikací.
- **irxevent** – daemon zasílající parametry v podobě eventů jednotlivým aplikacím jako klávesové zkratky atd.

Využívání mode a ovládání více programů stejnými tlačítky:

Pokud chceme ovládat stejnými tlačítky více programů, můžeme využít tzv. mody systému lirc. Po nastartování nějakého programu přepneme lirc do modu, který je spojený s konfiguračním souborem pro daný program.

Poté má každý program svůj konfigurační soubor, který je přepnutím se do s ním svázaného módu pomocí include připojený do hlavního souboru lircrc.

```
##########################
# XBMC
##########################
begin
prog   = irexec
remote = *
button = KEY_POWER
repeat = 0
delay  = 0
config = xbmc
mode   = xbmc
end
##########################
# MODES
##########################
begin xbmc
   include /etc/lirc/lircrcmode/xbmc
end xbmc
```

Kód 7: Příklad souboru lircrc s využitím irexec a mode

```
# Vypnutí xbmc
begin
prog   = irxevent
button = KEY_RED
config = Key s CurrentWindow
end
```

Kód 8: Externí soubor využívaný mode a irxevent

6.9 MySQL

MySQL je databázový systém, vytvořený švédskou firmou MySQL AB, později vlastněný společností Sun Microsystems a nyní Oracle Corporation. MySQL je jednou z nejznámějších a nejpoužívanějších databází vůbec.

Popis a zdůvodnění volby:

- Multiplatformní databáze dostupná na většině operačních systémů.
- Databáze vydávaná pod dvěma licencemi opensource GPL a komerční placené.
- Databáze optimalizovaná pro čtení (z tohoto důvodu velice využívaná pro webové aplikace).
- Výborná integrace s ostatními nástroji.
- Velké množství rozšiřujících enginů, pokrývajících široký záběr mnoha různorodých úloh.

Více v [14].

6.9.1 Architektura MySQL

Architektura MySQL se velmi odlišuje od architektur jiných databázových serverů, má široký záběr a je užitečná pro řešení mnoha různorodých úloh.

1. První vrstva obsahuje většinu potřebných nástrojů klient/server, které jsou založeny na síti.
2. Druhá vrstva obsahuje většinu mozku MySQL zajišťující (parsování, analýzu, optimalizaci). Veškerá funkcionalita, která se poskytuje prostřednictvím úložných enginů.
3. Třetí vrstva obsahuje úložné enginy. Ty mají na starosti ukládání a získávání všech dat uložených v MySQL. Server komunikuje s úložnými enginy prostřednictvím API úložných enginů. Toto rozhraní skrývá rozdíly mezi jednotlivými úložnými enginy a činí je na vrstvě dotazů velmi transparentními.

Více v [14].

7 Software využitý pro vývoj

7.1 Programování

- Fedora 13 (Goddard), Fedora 14 (Laughlin)
- Eclipse 3.6 (Helios), Eclipse 3.7 (Indigo)
- WindowBuilder Pro
- MySQL 5.1.56
- Apache 2.2.17
- phpMyAdmin 3.3.9
- MySQL Workbench 5.2

7.2 Grafická a textová část

- GIMP 2.6.11
- Violet UML editor 0.21.1
- LibreOffice 3.4.1

8 Software nutný pro běh

- Java
- GStreamer
- MySQL
- Operační systém (např. Linux)

9 Podrobný popis a implementace jednotlivých funkčních celků

9.1 Struktura aplikace

Aplikace je navržená tak, aby jí bylo možno umístit kdekoliv ve filesystému.

- Linux(Unix) – zde je ideální aplikaci umístit buď v **/home/user/GSJukeBox** adresáři uživatele, pod kterým bude aplikace spouštěna, nebo v adresáři **/opt/GSJukeBox**.
- Windows – zde je ideální umístění aplikace v adresáři **C:\Program Files\GSJukeBox**.

Dále bude popisovaná struktura a umístění pro operační systém Linux, jelikož na tomto systému bude aplikace prezentována. Předpokládejme umístění aplikace v adresáři „opt“ a spuštění aplikace pod uživatelem a skupinou jukebox (jukebox:jukebox).

Popis jednotlivých částí:

- **/opt/GSJukeBox** – umístění aplikace (RootApp).
- **/opt/GSJukeBox/conf/** – složka conf obsahuje konfigurační soubory aplikace a je nutné, aby vždy byla umístěna v rootu aplikace. Konfigurační soubory jsou načítány při startu aplikace z umístění (path/RootApp/conf/).
  - gsjukebox.properties
  - log4j.properties
- **/opt/GSJukeBox/icons/** – složka icons obsahuje ikony užívané v aplikaci a je nutné, aby vždy byla umístěna v rootu aplikace. Ikony jsou načítané při startu aplikace z umístění (path/RootApp/icons/).
- **/opt/GSJukeBox/logs/** – složka logs obsahuje aplikační log, cestu k tomuto souboru je nutné nastavit v log4j.propertis jako cesta k souboru pro file appender. (log4j.appender.log.File=/opt/GSJukeBox/logs/gsjukebox.log)
  - gsjukebox.log
- **/opt/GSJukeBox/data/** – složka data obsahuje vlastní fyzická data (mp3, obal alba, titulky) a může být umístěna v jakémkoliv místě filesystému. Je však nutné splnit několik podmínek:
  - cesta ke složce musí být nakonfigurována v gsjukebox.propertis (pathData=/opt/GSJukeBox/data)
  - musí zachovat určitou strukturu
    1. Ve složce data jsou umístěny složky interpretů.
    2. Složka interpreta obsahuje jednotlivá alba.
    3. Alba mohou obsahovat (muziku – song.mp3, obal alba – cover.jpg, titulky – song.srt).
- **/opt/GSJukeBox/gsjukebox.jar** – vlastní java aplikace.

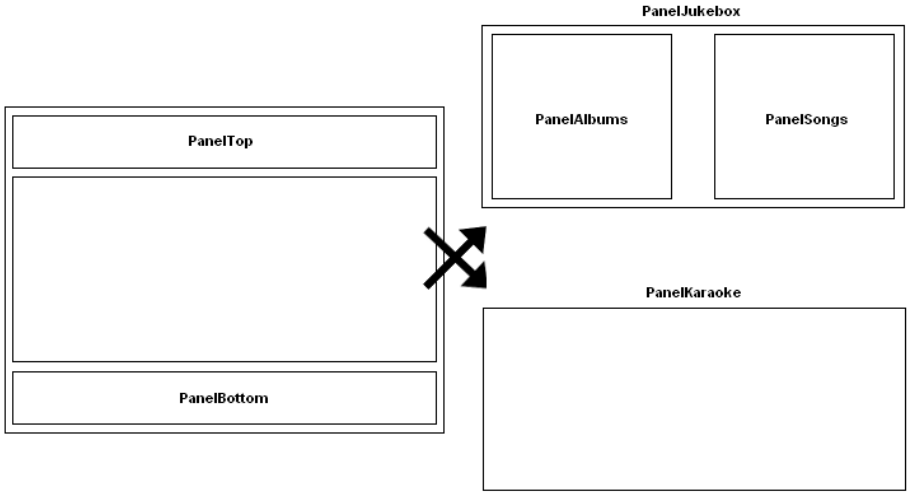
9.2 GUI

Pro tvorbu vlastního GUI jak již bylo popsáno, byl vybrán GUI toolkit Swing. GUI je rozčleněno do několika funkčních celků a každý tento celek tvoří pomyslnou komponentu (tyto komponenty jsou poděděny od komponenty JPanel a dále budou v textu zmiňovány jako **jukebox komponenty**). Každá z těchto pomyslných komponent je implementovaná dle návrhového vzoru Singleton (existence pouze jedné instance dané komponenty, jednoduchý přístup k instanci komponenty v celém kontextu aplikace). GUI dále obsahuje dialogy, které slouží pro komunikaci s uživatelem (potvrzení volby, zobrazení chybového stavu atd.).

Podrobný popis implementace:

1. package – **org.gsjukebox.gui**
   - **AppJukeBox.java** – obsahující stejnojmennou třídu sloužící pro spouštění a inicializaci celé aplikace.
   - **Methods.java** – obsahující pomocné statické metody (konvertovací, formátovací atd.).
   - **PanelTop.java** – horní komponenta obsahující menu jukeboxu.
   - **PanelBottom.java** – spodní komponenta obsahující zobrazení aktuálního stavu financí a aktuálního času přehrávaného songu.
   - **PanelJukebox.java** – tato komponenta v sobě spojuje komponenty PanelAlbum a PanelSong, aby se s nimi dalo pracovat jako s jedním celkem.
   - **PanelAlbums.java** – komponenta obsahující výpis alb.
   - **PanelSongs.java** – komponenta obsahující výpis songů vybraného alba a frontu songů určenou pro přehrání.
   - **PanelKaraoke.java** – komponenta zajišťující vizualizaci titulků při přehrávání karaoke.
2. package – **org.gsjukebox.gui.dialogs**
   - **DialogAlbumDetail.java** – dialog zobrazující detailní informace o vybraném albu.
   - **DialogError.java** – dialog zobrazující chybové, varující a informační zprávy.
   - **DialogPlaySong.java** – dialog sloužící jako rozhodující dialog pro přehrání muziky (potvrzení, zrušení – přehrát jako muziku, karaoke).
   - **DialogSongDetail.java** – dialog zobrazující detailní informace o vybraném songu.
   - **DialogStatistics.java** – dialog zobrazující statistiky o přehrávané muzice.
3. package – **org.gsjukebox.testdevel**
   - **DialogTestDevel.java** – pomocný dialog sloužící jednak při vývoji a jednak při poinstalačním testování. Lze pomocí něj testovat správnou funkci logování, simulovat přidávání peněz aniž by se musel využívat mincovník atd.
   - **TestLog4j.java** – obsahuje metodu pro testování logování, která je využita při testování logování v dialogu DialogTestDevel.

GUI je z těchto jednotlivých jukebox komponent skládáno na základě toho, co má být v danou chvíli právě zobrazeno. Horní a spodní panely jsou zobrazeny permanentně a uprostřed se zobrazuje PanelJukebox(PanelAlbums + PanelSongs), nebo PanelKaraoke pokud je právě přehráváno karaoke.

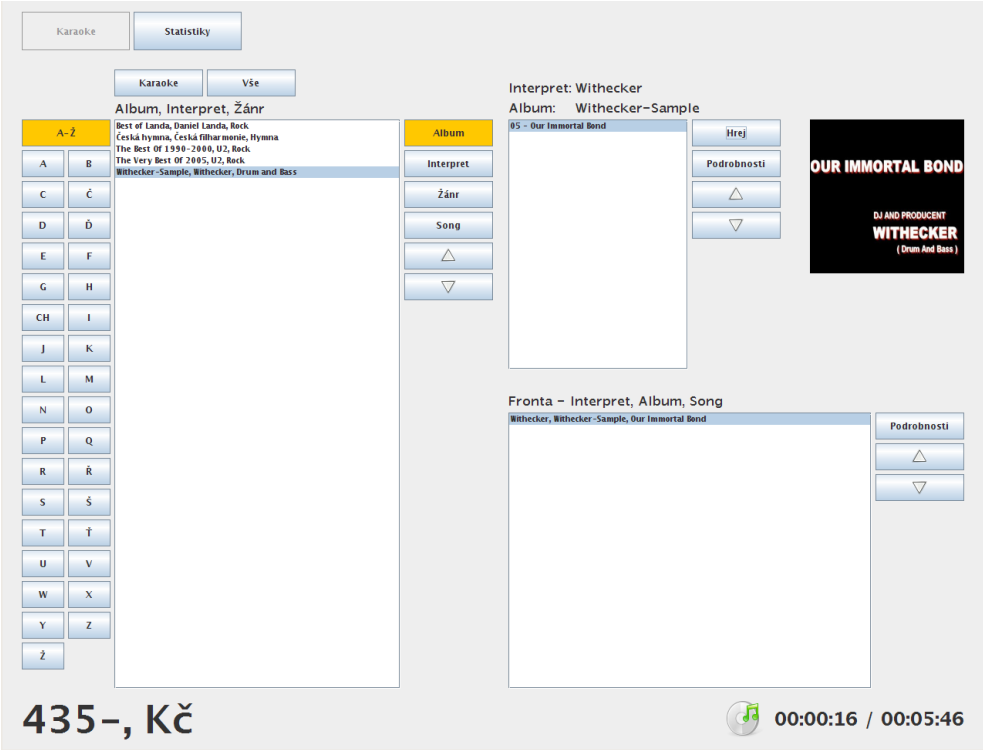


Obrázek 6: Skládání GUI z jukebox komponent

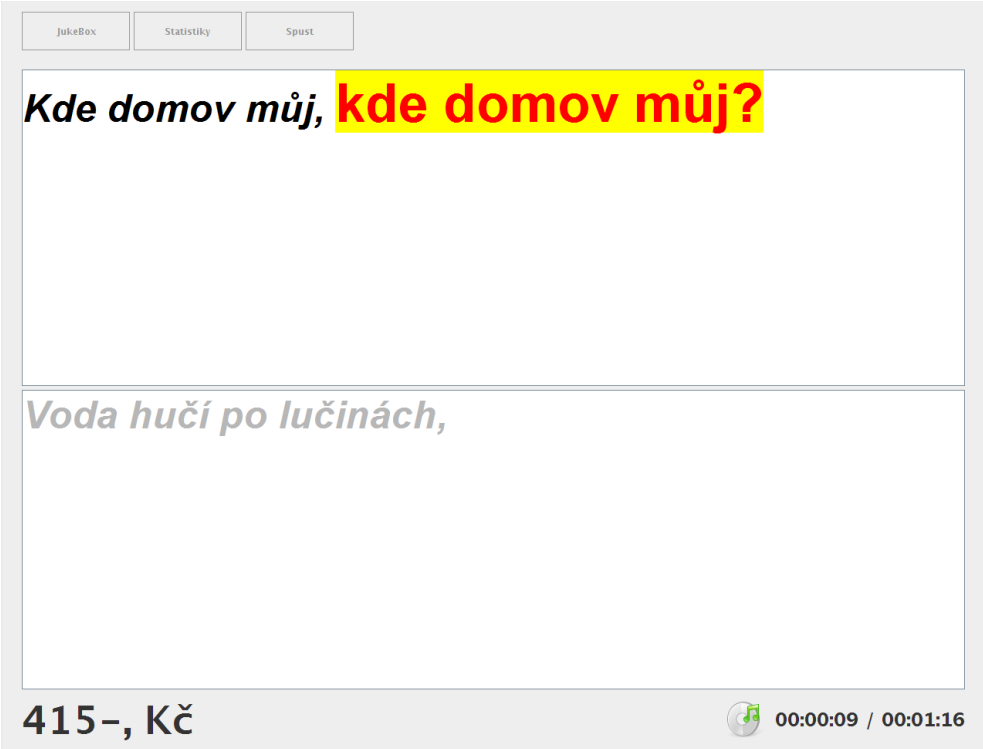
Jednotlivé jukebox komponenty dále implementují rozhraní **Runnable** a pomocí samostatného vlákna poskytují plynulý přechod mezi jednotlivými záznamy.

```
@Override
public void run() {
  while (true) {
    synchronized (this) {
        while (suspendFlag) {
            try {
                wait();
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
    //zdrojovy kod pro pohyb mezi daty
  }//while
}
private void suspend() {
        suspendFlag = true;
}
private synchronized void resume() {
        suspendFlag = false;
        notify();
}
```

Kód 9: Ukázka využití vlákna pro plynulý přechod mezi záznamy



Obrázek 7: Jukebox



Obrázek 8: Karaoke

9.2.1 Karaoke a titulky

Při přehrávání karaoke je potřeba zobrazovat text songu. Z důvodu jednoduchého časování pomocí nástrojů dostupných pro časování titulků, byl zvolen klasický formát filmových titulků (srt). Tento formát je rozšířen o dvojici znaků <>, které vymezují aktuálně zvýrazněný text.

Podrobný popis implementace:

1. package – **org.gsjukebox.subtitles**
   - **Subtitle.java** – tento soubor obsahuje stejnojmennou třídu, která reprezentuje objekt titulku. Obsahuje jednotlivé atributy titulku jako je jeho id, počáteční čas, koncový čas, text titulku, arraylist objektů **TextParsed.**
   - **TextParsed.java** – objekt reprezentující část textu titulku, obsahuje danou část text a informaci zda je zvýrazněná či nikoliv (false, true).
   - **SubtitleType.java** – tento soubor obsahuje stejnojmenou třídu, která obsahuje nadefinované konstanty třídy **SimpleAttributSet**. Třída SimpleAttributSet reprezentuje množinu atributů stylů, které je možné měnit.
   - **SubtitlesList.java** – tento soubor obsahuje stejnojmennou třídu, která jako parametr konstruktoru dostane soubor s titulky, tento soubor rozparsuje a vytvoří arraylist jednotlivých titulků. Tyto objekty typu **Subtitle** jsou pomoci metody insertSubtitleTextByClockTime, dle časů získaných z přehrávaného songu zobrazovány jako titulky karaoke.

```
static public final SimpleAttributeSet ACTUALCOLORED = new
SimpleAttributeSet();
static {
                StyleConstants.setForeground(ACTUALCOLORED, Color.red);
                StyleConstants.setFontFamily(ACTUALCOLORED, "Arial");
                StyleConstants.setBold(ACTUALCOLORED, true);
                StyleConstants.setFontSize(ACTUALCOLORED, 70);
                StyleConstants.setBackground(ACTUALCOLORED, Color.yellow);
 }
```

Kód 10: Příklad definice stylu pomocí třídy SimpleAttributSet

```
3
00:00:08,706 --> 00:00:12,137
Kde domov můj, <kde domov můj?>
```

Kód 11: Ukázka formátu titulků

9.2.2 MySQL a prezentace dat v GUI

Pro zobrazování dat jsou v jednotlivých jukebox komponentách využívány komponenty JList. Každý logický celek (album, song atd.) je reprezentován konkrétním objektem. Tyto objekty jsou vytvořeny z dat získaných prostřednictvím SQL dotazů a jsou vráceny jako kolekce objektů v podobě ArrayListu. Tyto objekty ArrayListu jsou poté vloženy jako jednotlivé záznamy do JListu. Kdykoli je tedy možné získat prostřednictvím konkrétního záznamu v JListu, konkrétní objekt a z něj konkrétní data. Objekty mají reimplementovanou metodu **toString()** ve které je nadefinováno jaká data a v jakem formátu se mají vypsat jako viditelná (**String**) položka JListu.

Podrobný popis implementace:

1. package – **org.gsjukebox.mysql**
   - **MySql.java** – tento soubor obsahuje stejnojmennou třídu, která obsahuje statické metody pro vracení dat z databáze. Data jsou vracena jako kolekce konkrétních objektů v podobě ArrayListu.
   - **Album.java** – tento soubor obsahuje stejnojmennou třídu, která reprezentuje položku album v (JList) seznamu alb.
   - **AlbumDetail.java** – tento soubor obsahuje stejnojmennou třídu, která reprezentuje objekt detail alba. Tento objekt obsahuje rozšiřující informace o albu (název alba, interpret, obal alba) a využívá se při zobrazení detailů o aktuálně vybraném albu.
   - **Song.java** – tento soubor obsahuje stejnojmennou třídu, která reprezentuje položku song v (JList) seznamu songů.
   - **SongDetail.java** – tento soubor obsahuje stejnojmennou třídu, která reprezentuje objekt detail songu. Tento objekt obsahuje rozšiřující informace o songu (název songu, pořadí na albu, žánr atd.) a využívá se při zobrazení detailů o aktuálně vybraném songu.
   - **QueueSongs.java** – tento soubor obsahuje stejnojmennou třídu, která reprezentuje položku song v (JList) frontě pro přehrávání.
   - **Statistics.java** – tento soubor obsahuje stejnojmennou třídu, která reprezentuje položku song ve (JList) statistikách.

```
public class Album {

	private int id_album;
	private String album;

	public Album(int id_album, String album){
		this.id_album = id_album;
		this.album = album;
	}
	public int getId_album() {
		return id_album;
	}
	public String getAlbum() {
		return album;
	}
	@Override
	public String toString() {
		return album;
	}
}
```

Kód 12: Ukázka objektu reprezentující album

```
public static ArrayList<Album> getAlbums(){

	//implementace kodu

	return arr;
}
```

Kód 13: MySQL.java – metoda vracející <Album>ArrayList

```
private DefaultListModel defaultListModelAlbums = new DefaultListModel();
public void insertArrAlbumsToList(ArrayList<Album> arrayList){

	defaultListModelAlbums.clear();
	for(int i=0; i<arrayList.size(); i++){
		defaultListModelAlbums.addElement(arrayList.get(i));
	}
	listAlbums.setSelectedIndex(0);
	}
}
```

Kód 14: Vložení dat z <Album>ArrayList do JList

9.3 GStreamer – přehrávání

Z důvodu zvolení GUI toolkitu Swing není zapotřebí knihovna pro práci s SWT (gstreamer-java-swt).

Podrobný popis implementace:

1. package – **org.gsjukebox.audio**
2. knihovny
   - **gstreamer-java-1.5.jar**
   - **jna-3.2.4.jar**
3. **PlayerAudio.java** – tento soubor obsahuje stejnojmennou třídu implementovanou dle návrhového vzoru singleton pro zajištění existence vždy pouze jedné instance pracující s daným multimediálním frameworkem GStreamer. Ze samotného frameworku je využívána třída **PlayBinMediaPlayer** spolu s posluchačem událostí **MediaListener**. MediaListener poskytuje metody reagující na jednotlivé události potřebné k řízení přehrávání muziky. Tyto metody lze jednoduše přepsat (override) a naprogramovat zde vlastní potřebnou funkcionalitu. Dále jsou zde naprogramovány metody umožnující přehrávání muziky i karaoke s potřebnými pomocnými konfiguračními metodami.

```
private PlayBinMediaPlayer player = new PlayBinMediaPlayer();
player.setVideoSink(ElementFactory.make("fakesink", "videosink"));
player.addMediaListener(new MediaListener() {
        @Override
        public void stop(StopEvent arg0) {
        }
        @Override
        public void start(StartEvent arg0) {
        }
        @Override
        public void positionChanged(PositionChangedEvent arg0) {
        }
        @Override
        public void pause(StopEvent arg0) {
        }
        @Override
        public void endOfMedia(EndOfMediaEvent arg0) {
        }
        @Override
        public void durationChanged(DurationChangedEvent arg0) {
        }
});
```

Kód 15: PlayBinMediaPlayer a MediaListener

9.4 Properties – konfigurace aplikace

V aplikaci jsou využívané dva properties soubory:

1. log4j.properties – který je podrobněji popsán v následující kapitole.
2. gsjukebox.properties – obsahující konfigurační parametry aplikace GSJukebox.

Oba soubory jsou umístěny v adresáři conf, který musí být vždy v hlavním adresáři aplikace. Soubor gsjukebox.properties obsahuje konfigurační parametry celé aplikace. Zvolený formát properties souboru je **key=value**.

Podrobný popis implementace:

1. package – **org.gsjukebox.params**
2. **ConfProperties.java** – tento soubor obsahuje stejnojmennou třídu implementovanou dle návrhového vzoru singleton pro zajištění existence vždy pouze jedné instance pracující s daným konfiguračním souborem gsjukebox.properties. S **ConfProperties** se dále pracuje v hlavním objektu **ParamsConfig**, který obsahuje všechny konfigurační parametry aplikace jako statické proměnné tak, aby se s nimi lehce pracovalo v celém systému. Obsahuje jak konfigurační proměnné ze souboru **gsjukebox.properties**, tak i konfigurační parametry mincí z konfigurační tabulky **tab_money** a parametry nastavované při kompilaci systému.

```
################################################
# percent of screen x,y (e.g 100,100)
################################################
widthDialogStatistics=100
heightDialogStatistics=100

################################################
# gui test devel mode - true/false
################################################
guiTestDevelMode=true

################################################
# path to data
################################################
pathData=/home/migi/GSJukeBox/data

################################################
# dev - coinselector - /dev/ttyUSBX
################################################
devCoinSelector=/dev/ttyUSB0
```

Kód 16: Ukázka gsjukebox.properties

9.5 Log4J – logování

Podrobný popis implementace:

1. package – **org.gsjukebox.log**
2. knihovna – **log4j-1.2.16.jar**
3. **LoggerLog4j.java** – tento soubor obsahuje stejnojmennou třídu implementovanou dle návrhového vzoru singleton pro zajištění existence vždy pouze jedné instance reprezentující daný logger. Třída LoggerLog4j obsahuje pouze vytvoření loggeru a načtení properties souboru.
4. **log4j.properties –** v aplikaci je zvolena možnost konfigurace všech potřebných části v properties souboru. Aplikace není poté nutno rekompilovat po každé změně v konfiguraci log4j (logger, appender, layout).

```
PropertyConfigurator.configure(System.getProperty("user.dir") +
"/conf/log4j.properties");
Logger logger = Logger.getLogger(LoggerLog4j.class);
```

Kód 17: Vytvoření instance loggeru a načtení properties file

```
# level logger - ALL, INFO, ERROR, DEBUG, OFF
# appenders - stdout, log
log4j.rootLogger=ALL, stdout, log

#############################################################
# appenders
#############################################################
# stdout
log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.Target=System.out
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern=[%p] - %d (%F:%M:%L) - %m
%n

# log
log4j.appender.log=org.apache.log4j.FileAppender
log4j.appender.log.File=/home/migi/GSJukeBox/logs/gsjukebox.log
log4j.appender.log.layout=org.apache.log4j.PatternLayout
log4j.appender.log.layout.ConversionPattern=[%p] - %d (%F:%M:%L) - %m%n
```

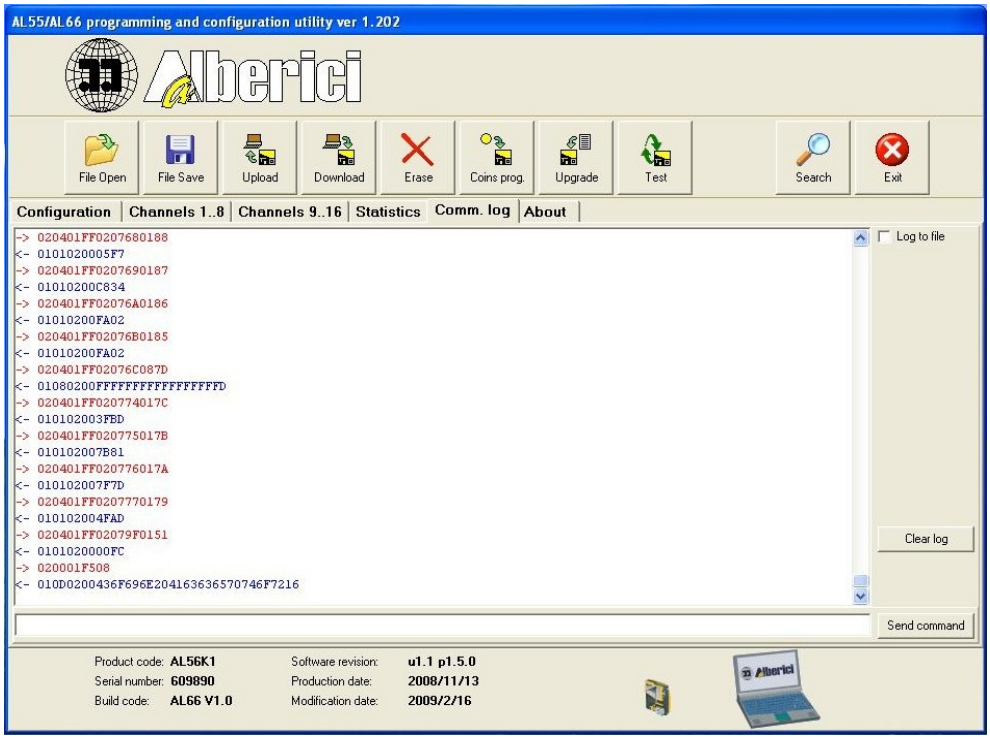
Kód 18: log4j.properties

9.6 ccTalk – mincovník

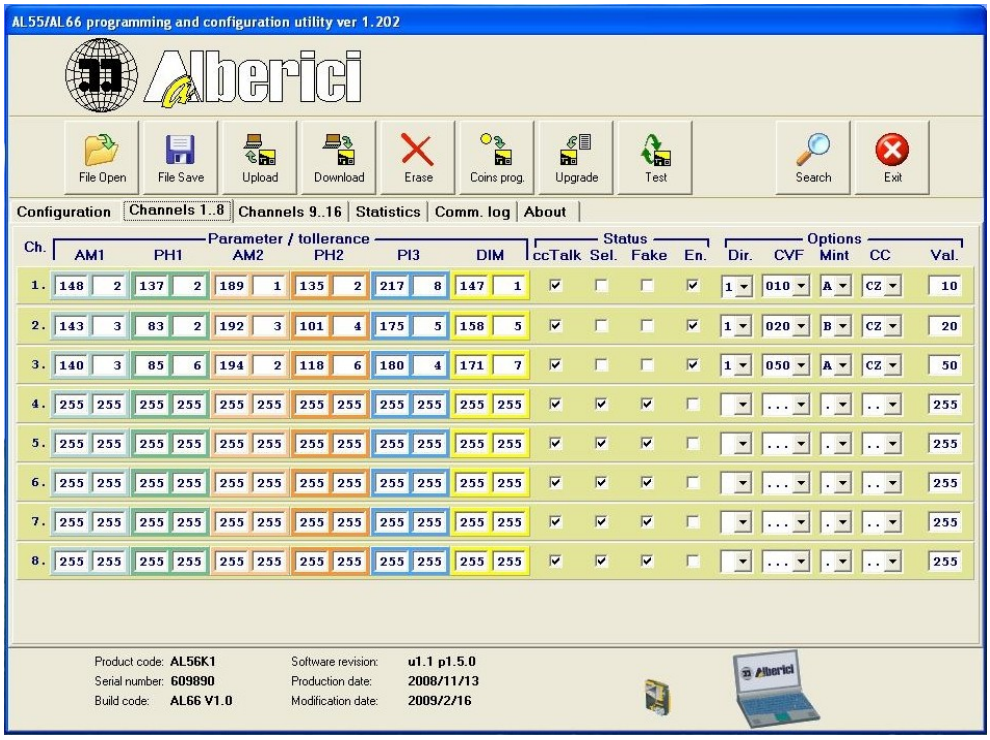
9.6.1 Programování mincovníku

Zvolený mincovník je dodávaný s programovací a konfigurační utilitou, pomocí které lze mincovník nakonfigurovat a naprogramovat pro přijímání různých druhů mincí. Tento nástroj je k dispozici pouze pro operační systémy Microsoft Windows, a tudíž je zapotřebí využít Windows, Wine nebo Linux s virtualizovanými Windows. Právě tato poslední možnost byla využita.

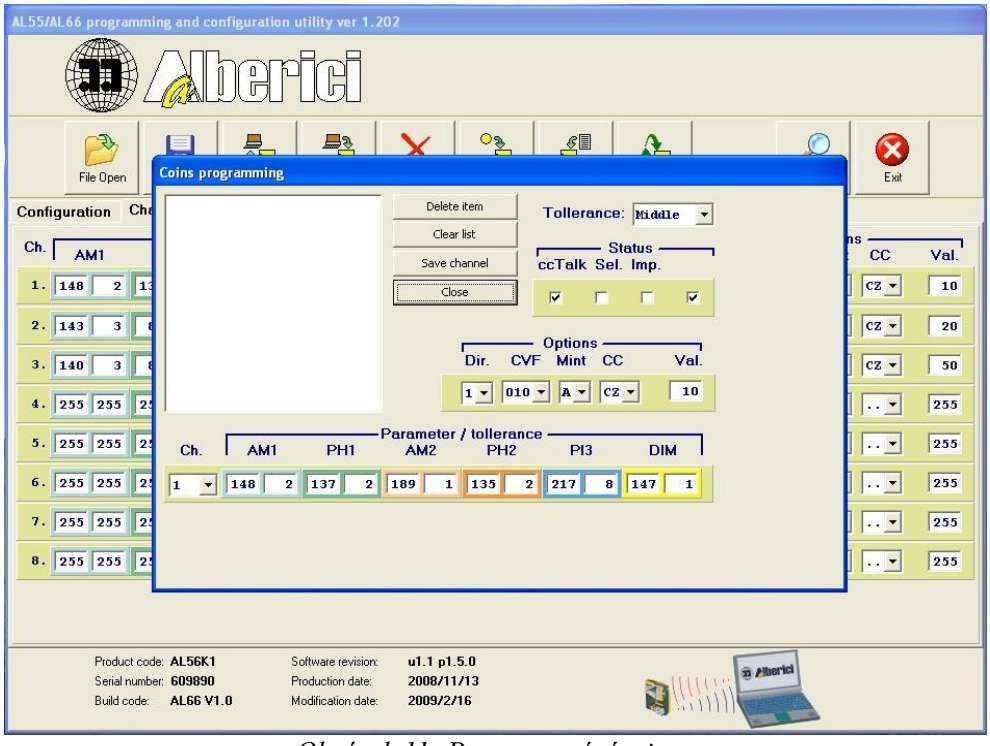
- Vlastní programování probíhá pomocí USB interface.
- Mincovník podporuje několik typů validace.
- Celkem lze současně validovat 16 různých mincí prostřednictvím 16 kanálů.
- Mincovník podporuje pokročilé funkce jako statistiky (počet vhozených mincí, počet akceptovaných a zamítnutých mincí atd.).
- Comm. Log – je konzole umožňující zasílat mincovníku TX zprávy a prijímat RX zprávy pro simulování a testování komunikace. Uživatel nemusí zasílat celou zprávu, ale pouze hlavičku, popřípadě datovou část. Ostatní části včetně checksum doplní konfigurační utilita.



Obrázek 9: Comm. Log – konzole



Obrázek 10: Naprogramované kanály 1-3



Obrázek 11: Programování mince

9.6.2 ccTalk – vlastní implementace komunikace v Jave

V reprezentaci sériového rozhraní se operační systémy Windows a Linux(Unix) liší natolik, že je zde nutná rozdílná implementace. Operační systémy od firmy Microsoft reprezentují sériové rozhraní jako **COMX** zařízení, zatímco Linux(Unix) systémy jako zařízení **/dev/ttySX**, popřípadě USB sériové zařízení **/dev/ttyUSBX**.

Existují dvě možnosti:

1. Dvě různé implementace pro systémy Windows a Linux(Unix).
2. Využití emulace systému Linux(Unix) na systémech Windows pomocí nástroje Cygwin.

Dále bude popisovaná implementace právě pro konkrétní operační systém, na kterém bude systém prezentován (GNU/Linux).

Podrobný popis implementace:

1. package – **org.gsjukebox.coinselector**
2. package obsahuje 4 java soubory:
   - **CoinSelector.java** – soubor obsahuje třídu **CoinSelector**, která implementuje rozhraní Runnable a zajišťuje v samostatném vláknu komunikaci s mincovníkem. Třída je implementovaná dle návrhového vzoru singleton a obsahuje metodu MsgWriteRead, ve které probíhá zasílání TX a přijímání RX zpráv.
   - **Header.java** – soubor obsahuje **enum Header**, který obsahuje hlavičky zpráv cctalku. Header neobsahuje všechny hlavičky cctalku, pouze ty, jež jsou využívány pro komunikaci s mincovníkem.
   - **MsgTx.java** – obsahuje třídu MsgTx reprezentující TX zprávu a metody pro práci s jeho daty.
   - **MsgRx.java** – obsahuje třídu MsgRx reprezentující RX zprávu a metody pro práci s jeho daty.
   - **NumSys.java** – obsahuje třídu NumSys s pomocnými metodami pro konvertovaní z int do unsigned byte a z unsigned byte do int (intToUnsignedByte, unsignedByteToInt). Datový typ **byte** v Jave reprezentuje číslo -127 až +128, proto je nutné pomocí bitového operátoru AND (&) z konvertovat číslo 0 až 256 na číslo -127 až +128.

```
@Override
public void run() {
      Runtime.getRuntime().exec("stty -F " + dev + " 9600 min 0 cs8 -parenb
-cstopb");
      dis = new DataInputStream(new FileInputStream(dev));
      dos = new DataOutputStream(new FileOutputStream(dev));

      MsgTx msgTx = null;
      while(true){
        msgTx = new MsgTx(Header.ReadBufferedCreditOrErrorCodes);
        LoggerLog4j.getInstance().getLogger().debug(msgTx.getDecimal());
        LoggerLog4j.getInstance().getLogger().debug(msgTx.getByte());
        msgRx = this.MsgWriteRead(msgTx);
        LoggerLog4j.getInstance().getLogger().debug(msgRx.getDecimal());
        LoggerLog4j.getInstance().getLogger().debug(msgRx.getDataAscii());
        coin = msgRx.getNumberOfCoin();
      }//while
}//run
```

Kód 19: Ukázka komunikace s mincovníkem

```
private MsgRx MsgWriteRead(MsgTx msgTx){
      MsgRx msgRx = null;
      byte bufRx[] = new byte[200];
      int sizeRx = 0;
      try {
            dos.write(msgTx.toByteArray());
            Thread.sleep(100);
            sizeRx = dis.read(bufRx);
            msgRx = new MsgRx(bufRx, sizeRx);
            LoggerLog4j.getInstance().getLogger().debug("Pocet prectenych byte: "
+ sizeRx);
      } catch (IOException e) {
        //write, read
        e.printStackTrace();
      } catch (InterruptedException e) {
        // Thread.sleep
        e.printStackTrace();
      }
      return msgRx;
}
```

Kód 20: Metoda MsgRx MsgWriteRead(MsgTx msgTx)

9.7 LIRC – dálkový ovladač

Je nanejvýš vhodné, aby bylo možné regulovat hlasitost jukeboxu prostřednictvím dálkového ovládání a obsluha nemusela neustále chodit regulovat hlasitost k zařízení samotnému.

Co je nutné ovládat:

- Mutování zvuku
- Zesilování zvuku
- Zeslabování zvuku

Následující konfigurace systému lirc je již popisována na operačním systému, na kterém bude jukebox prezentován (GNU/Linux).

9.7.1 PulseAudio

PulseAudio je v současnosti standardním systémem pro ovládání zvuku na linuxových systémech. Výhodou je integrace ovládání přes lirc systém, díky čemuž nebudeme potřebovat programy irexec a irxevent.

1. Nejprve je nutné nakonfigurovat pulseaudio pro práci s lirc systémem. Pomocí **pulseaudio --dump-modules** vypíšeme veškeré moduly pulseaudia, které lze načíst.
2. V konfiguračním souboru pulseaudia **/etc/pulse/default.pa** přidáme načtení modulu pro ovládání přes lirc systém **load-module module-lirc**.
3. Vygenerujeme konfigurační soubor **lircd.conf** s názvy tlačítek a jejich číselnými kódy.
4. Vytvoříme soubor **lircrc** s namapovanými akcemi na jednotlivé názvy tlačítek.
5. Spustíme daemon **lircd** s potřebnými parametry. Spouštění při startu systému můžeme zajistit přidáním startovacího příkazu do souboru /etc/rc.d/rc.local.

```
lircd -H dev/input -d /dev/input/eventX
```

Kód 21: Příklad spouštění daemona lircd

```
# Zvuk mute
begin
   prog = pulseaudio
   remote = *
   button = KEY_MUTE
   config = mute-toggle
end
```

Kód 22: Příklad souboru lircrc s namapovanými akcemi pulseaudia

10 Licence hudební produkce – OSA a INTERGRAM

O licencování hudební produkce pro zařízení jako je jukebox se starají v ČR dva kolektivní správci OSA a INTERGRAM. Jejich působnost se liší okruhy těch které zastupují.

1. OSA – zastupuje autory hudby a textů k hudbě a hudební nakladatelé.
2. INTERGRAM – zastupuje výkonné umělce, výrobce zvukových a zvukově-obrazových záznamů.

10.1 Jukebox

Jukeboxem se rozumí přístroj hudební automat s mincovníkem nebo bez mincovníku, umožňující užití předmětů ochrany, a to zcela ze zvukových nebo zvukově obrazových záznamů, jejichž výběr a pořadí jsou obvykle ovlivňovány. K přístroji je možné připojit nejen reproduktory, ale i monitory umožňující přenos obrazové složky díla.

10.2 Licence

Na základě smlouvy uzavřené mezi OSA a INTERGRAM o pověření zastupováním při výkonu kolektivně spravovaného práva na provozování uměleckých výkonů zaznamenaných na zvukový záznam a takovýchto záznamů prostřednictvím jukeboxů a podobných přístrojů, lze platit pouze jeden poplatek, a tito dva kolektivní správci si danou finanční odměnu vyúčtují mezi sebou.

10.3 Poplatek

Výše odměny je uvedena ve smlouvě. V tabulce je uvedena orientační výše poplatků licencování hudební produkce pro zařízení jako je jukebox.

Druh přístroje	Měsíční odměna
audio jukebox	1.628,- Kč
video jukebox	2.805,- Kč

Tabulka 10: Orientační výše poplatků

Více v [15].

11 Závěr

11.1 Zhodnocení

Cílem práce bylo vytvořit komplexní řešení multiplatformního jukeboxu. Toto řešení je složeno jak z HW tak SW části, přičemž u SW části byl požadavek na multiplatformnost řešení s důrazem na využití OSS technologií. V drtivé většině jsou OSS technologie synonymem multiplatformních technologií, tudíž se tyto dvě podmínky podařilo splnit velice dobře.

OSS technologie dokazují, že jsou to velice kvalitní technologie, které lze využít stejně dobře pro „enterprise“ aplikace, jako i pro „standalone“ řešení (jukebox, informační kiosek, atd.). V obou případech OSS technologie usnadňují práci, díky množství dostupných knihoven a materiálů (manuály, tutoriály atd.) a tak jednoznačně minimalizují náklady na vyvíjený systém.

Přestože je drtivá většina výsledné aplikace multiplatformní, existují zde malé části implementace platformně závislé. Příkladem platformně závislé části je samotná komunikace s mincovníkem přes sériový port. Tato komunikace je rozdílná, z důvodu rozdílné reprezentace sériových portů a práce s nimi na operačních systémech Linux(Unix) a operačních systémech od firmy Microsoft. V těchto případech je zvolena implementace pro systém GNU/Linux, protože na tomto operačním systému bude jukebox prezentován.

11.2 Budoucí vývoj

Pro budoucí vývoj se naskýtá nepřeberné množství rozšíření:

- Implementace vlastního Swing LAF, pro zlepšení grafické vizáže jukeboxu.
- Přidání schopnosti přehrávat videoklipy.
- Další možností je implementace přehrávání muziky z různých internetových zdrojů.
- atd.

12 Literatura

[1] STEIN, R. *Návrh aplikací v jazyce UML* [online]. Interval.cz, 2005 [cit. 03-05-2008]. Dostupný z WWW: <http://interval.cz/clanky/navrh-aplikaci-v-jazyce-uml-diagram-trid/>.

[2] WIKIPEDIE. *Wikipedie* [online]. Wikipedie, 2008 [cit. 03-05-2008]. Dostupný z WWW: <http://cs.wikipedia.org>.

[3] HEROUT, P. *Učebnice jazyka Java*. 1. dotisk vyd. České Budějovice : KOPP, 2004. 349 s. ISBN 80-7232-115-3.

[4] HEROUT, P. *JAVA – bohatství knihoven*. 1. vyd. České Budějovice : KOPP, 2003. 242 s. ISBN 80-7232-209-5.

[5] ECLIPSE.ORG. *SWT: The Standard Widget Toolkit* [online]. Eclipse.org, 2005-2011 [cit. 31-07-2011]. Dostupný z WWW: <http://www.eclipse.org/swt/>.

[6] ROSEINDIA.NET. *Java Swing Tutorials* [online]. Roseindia.net, 2008 [cit. 31-07-2011]. Dostupný z WWW: <http://www.roseindia.net/java/example/java/swing/>.

[7] ROSEINDIA.ORG. *Java get set Properties* [online]. Roseindia.net, 2008 [cit. 01-07-2011]. Dostupný z WWW: <http://www.roseindia.net/java/java-get-example/java-get-set-properties.shtml>.

[8] ROSEINDIA.NET. *log4j* [online]. Roseindia.net, 2009 [cit. 31-7-2011]. Dostupný z WWW: <http://www.roseindia.net/tutorials/log4j/>.

[9] GSTREAMER.FREEDESKTOP.ORG. *GStreamer* [online]. Gstreamer.freedesktop.org, 2011 [cit. 31-07-2011]. Dostupný z WWW: <http://gstreamer.freedesktop.org/>.

[10] GSTREAMER-JAVA. *gstreamer-java* [online]. GStreamer-java, 2011 [cit. 31-07-2011]. Dostupný z WWW: <http://code.google.com/p/gstreamer-java/>.

[11] WIKIPEDIA.ORG. *CcTalk wiki* [online]. Wikipedia.org, 2011 [cit. 31-07-2011]. Dostupný z WWW: <http://en.wikipedia.org/wiki/CcTalk>.

[12] MONEY CONTROLS. *ccTalk Serial Communication Protocol* [online]. Money Controls, 2011 [cit. 31-07-2011]. Dostupný z WWW: <http://www.moneycontrols.com>.

[13] LIRC.ORG. *LIRC - Linux Infrared Remote Control* [online]. Lirc.org, 2011 [cit. 31-07-2011]. Dostupný z WWW: <http://www.lirc.org/>.

[14] WIKIPEDIE.ORG. *MySQL* [online]. Wikipedie.org, 2011 [cit. 07-08-2011]. Dostupný z WWW: <http://cs.wikipedia.org/wiki/MySQL>.

[15] OSA. *Jukeboxy a podobné přístroje* [online]. OSA, 2011 [cit. 14-08-2011]. Dostupný z WWW: <http://www.osa.cz/>.

13 Obsah CD

- /doc/obr/db/**db_jukebox_er_diagram.png** – obrázek struktury databáze.
- /doc/obr/other/***.*** – ostatní obrázky (použité v dokumentaci, obrázky jukeboxu atd.).
- /doc/lirc/***.*** – příklad konfiguračních souborů LIRC.
- /doc/doc_install_conf/**doc_install_conf.odt** – instalační a konfigurační příručka ve formátu odt (ODF).
- /doc/doc_install_conf/**doc_install_conf.pdf** – instalační a konfigurační příručka ve formátu pdf.
- /sql/**db_jukebox.sql** – skript pro vytvoření databáze db_jukebox.
- /sql/**db_user.sql** – skript pro vytvoření uživatele pro práci s databází db_jukebox.
- /src/**src.zip** – zdrojové kódy java aplikace.
- /text/**jukebox.odt** – textová část diplomové práce ve formátu odt (ODF).
- /text/**jukebox.pdf** – textová část diplomové práce ve formátu pdf.
- **GSJukeBox.tar** – kompletní aplikace.

14 Přílohy

Přílohy jsou umístěny na CD v adresáři **doc**.

I. Obrázek struktury databáze a ostatní obrázky.

II. Příklad konfiguračních souborů systému LIRC.

III. Instalační a konfigurační příručka.